Contact Planning for Acyclic Motion with Tasks Constraints

Adrien Escande and Abderrahmane Kheddar

Abstract—This paper extends our previous work on contact points planning in two ways. First, by taking advantage of the possibilities offered by our initial posture generator, we include additional tasks that are not related to locomotion within the planning. The output motion will be generated so as to cope with these tasks. Second, we refine the potential function that guides the planner by introducing potential fields acting each on a single body. This helps escaping local minima of the original potential field and thus to deal with more challenging scenarios. We then test these novelties on difficult problems with success, and experiment the output of one of the planned scenario on a HRP-2 humanoid robot.

I. INTRODUCTION

Humanoids are anthropomorphic advanced robotic systems that are designed to mimic human functions in either collaborative (with humans) or standalone modes. Their highly redundant structure allows, in principle, to design rich motion behaviors and precise tasks in environments and application working contexts where they can apply. Despite this redundancy and the advanced hardware achievements, they are still in the infancy stage of their full exploitation and capabilities. This is because the problems of integration, fast planning, active perception and reactive behaviors are (among others) generally well understood but yet highly complex and unsolved. One of the prime functionality of humanoids is transportation motion which allows the robot to go from a place to another. To tackle this problem, very robust cyclic walking patterns, even with reactive capabilities, exist. The ASIMO humanoid robot, and recently Toyota’s partner humanoid can even run. Walking on uneven terrain has also been splendidly demonstrated for the quadrupedal BigDog robot; such demonstration can not be achieved yet on a humanoid.

Cyclic motions, although useful in practice, prove to be limited and there are plenty of situations where acyclic motion with eventually additional links’ support are required to achieve complex transportation motions. A simple example could be the humanoid robot grasping a ramp to ease climbing high stairs. This has been demonstrated in [1]. Non-gaited motion planning has been more formally addressed in simulation in [2] and [3]. A different approach has been proposed by the authors in [4] and [5] where real experiments have been conducted on the HRP-2 humanoid robot. Acyclic motions having contact supports that could occur on any part of the robot with any part of the environment is on the stage of full realization, but there remain many problems to be solved and future extension to be addressed that have been discussed in [6].

In this paper, we show that our approach allows also an interesting extension at nearly no cost to handle additional constraints related to tasks that are not related to locomotion. For instance, assume that we want not only to plan the sequence of supporting contact points between which trajectories are generated to go from a given location to another, but also to hold a container full of liquid --such as a cup of Japanese green tea-- and keep it with a constant orientation all along the motion. We present a method to realize such scenario in nearly straightforward way since our planner relies on an optimization-based posture generator that can be seen as a generalized inverse kinematics under constraints. Once the contact sequences are planned, ideally, the trajectories are generated from an optimization software (e.g. [7]) and can be played with a stack-of-tasks reactive controller such as the one proposed in [8] or in [9].

We also present an improvement of the way we build the potential field that guides the planner. This makes the robot more likely to take larger steps when possible, resulting in a faster planner. It also allows to deal efficiently with the local minima that inevitably come with important changes in the overall posture (for example, when going from a two-legged locomotion to a four legged one). Such changes are required in difficult environment.

Firstly, we quickly recall the background of our contact planning method, however we strongly recommend the reader to refer to our previous published work in [4] and [5]. This is followed by the way additional tasks constraints can be added and dealt with during planning. We then describe the improvement in the way of building the potential field. After it, the proposed ideas are exemplified through several scenarios including one where the HRP-2 is asked to bring a glass from some location, go toward the table, sit on the chair at this table, while keeping the glass of water vertically. Finally, this experiment is realized by the HRP-2 humanoid robot.

II. BACKGROUND

A. Overall planner

In [4] we presented a planner with the following principle: planning is made in the space of sets of contacts $SC$, by building incrementally a tree of such sets. The difference between a father and its son is exactly one contact (one more or one less). To drive the planning, a potential function $f$ is given. At each iteration, the best leaf of the tree (according to $f$) is selected and its sons are built by adding or removing a contact. If some of the new leaves are too close to existing nodes or leaves, they are discarded. This mechanism is inspired by the potential-field-based planner Best First Planning (BFP), see [10]. However, we are planning here in $SC$, which allows a dramatic reduction of the search
space compared to the usual configuration space. Yet it does not allow to take into account the geometrical and physical limitations of the robot: two contacts of a set may be too far from each other, a contact may force collisions or instability of the robot, etc. Feasibility of a set must be checked, and this is done with a posture generator (section II-B), see Fig. 1. Upon failure of the posture generator, the set of contacts is discarded.

The Posture Generator is a powerful module of our planner. It is used for projections on sub-manifolds of the configuration space as several other methods (see [14] and [15]), but in a very meaningful way, since the user has a great control on the projection method, by defining both the constraints and the optimization criteria. So far, we did not use its full potential during planning: we restricted the constraints to express contacts, stability, robot limits and collision avoidance. Yet functions $g_i$ and $h_i$ defined in (6) and (7) can be of any kind, provided they are twice continuous. We can thus write many other types of constraints than those of contacts $C_i$.

Equality constraints $g_j(q) = 0, j \in [1,j]$ define a sub-manifold on which the posture we are looking for must be. Such a set is called task in [14]. It is indeed really close to the notion of task function defined by Samson in [16], which relates to regulating the distance to a sub-manifold.
This notion of regulation is the only difference between both definition. This difference encompasses and allows to take into account the discrepancy between the planning in a perfect world with supposedly flawless models and the execution of the plan in a real environment.

Yet we are also using inequalities \( h_j(q) \leq 0 \) that restrict this sub-manifold to one of its subsets. We will call task, as a generalization of [14], a set of equalities and inequalities. Contact is a particular kind of task. Having the parallel in mind with Samson’s tasks is interesting because we can translates easily our tasks to be executed by a stack-of-tasks ([8]).

With these remarks, we can rewrite (6) and (7) in a more general way

\[
\begin{cases}
    g_i(q) = 0 & \forall \mathcal{T}_i \\
    h_i(q) \leq 0 & \forall \mathcal{T}_i
\end{cases}
\]

(8)

(9)

The \( \mathcal{T}_i \) being tasks.

Through the addition of tasks to the Posture Generator, inclusion of tasks may be then done in our planner in a very natural and easy way that does not need any particular coding effort, but the interface to insert these tasks in the posture generation problem related to a new node: for each node, the posture generator will solve the problem 1, with the equations of the tasks \( \mathcal{T}_i \) as additional constraints.

Here are some tasks that may be of great interest in a motion planning:

- maintaining the orientation of a carried object. We will give an example later,
- keep looking at a target, or maintaining visual features in the field of view. The latter can be useful, for example, for self-localization,
- looking at the contact that is being created, so that, once the plan is executed on the real robot, there will be a visual feedback, especially in the absence of contact sensors, etc.

Let’s take as an example the following mission: the robot must carry a glass containing a liquid. If the robot does not have the glass in the gripper, it must first reach it. This part of the planning was solved in [4]. Separating the problem into sub-missions as “go to the glass”, “grasp the glass”, “carry it”, and so on, would be the purpose of an higher-level task planner and is beyond the scope of this work. Here we assume the robot with the glass grasped. In order not to spill a drop of liquid, the robot must keep at all time the glass in a vertical position, and in particular at each witness posture associated to a node of our planner. This task can be described with:

\[
\mathcal{T}_{\text{glass}} = \begin{cases}
    \mathbf{n}(q).i = 0 \\
    \mathbf{n}(q).j = 0 \\
    -\mathbf{n}(q).k < 0
\end{cases}
\]

(10)

where \( \mathbf{n}(q) \) is the axis of the hand carrying the glass, and \((i,j,k)\) is a frame of the world (cf Fig. 2). This task is added to every posture generation. Every posture of the output plan will respect its constraints. Adding a task in the planning decreases the number of degree of freedom of the robot, meaning a witness posture will be found for fewer sets of contacts. It has thus a direct impact on the plan, since it reduces the possibilities of the planner.

IV. IMPROVED POTENTIAL FIELD BASED ON GLOBAL TRAJECTORY GUIDE

This section presents a refinement in the way we build the potential field guiding the planning. This refinement aims at reducing local minima, making it possible for the planner to tackle more advanced scenarios.

In potential field based planning local minima are a major cause of inefficiency, even if there are methods to escape from them. In classical potential field based planning (see [10]), local minima arise from the repulsive fields of the obstacles. In contact point planning however, obstacles are not to be avoided because the robot may need them as supports (even the floor is then considered as an obstacle in the workspace). We can not have potential fields that drive the robot away from them. We only have an attractive potential function toward the goal. Yet while evolving in this potential field, the robot may be blocked by some obstacles which, de facto create local minima (just as inequality constraints in optimization), for example when there is an obstacle in the steepest descent direction. These are often huge local minima for which practical escape is impossible. To avoid these minima, we use a rough trajectory \( T \), that acts as a guide to go around obstacle, as well as giving information about the preferred posture at each location. Upon this guide, we construct a descending valley-like potential field over the configuration space, as depicted in Fig. 3 and explained

![Fig. 2. Vectors linked to the definition of task \( \mathcal{T}_{\text{glass}} \).](image)

![Fig. 3. Based on a trajectory in the configuration space (in red) which acts as a trajectory of reference postures, we build a descending valley-like potential by combining the distance to the trajectory and the distance traveled along the trajectory.](image)
in [5]. Roughly speaking this potential field embed two terms: the distance \( d \) from the current configuration to the trajectory \( T \), and a measure of the distance \( l \) traveled along the trajectory.

\[
f_T(q) = d(q, T)^2 - k \cdot l(q)
\]  

(11)

\( T \) thus acts as trajectory of reference postures. While this copes with relatively big local minima and proved to be effective, we do not tackle some smaller ones, that are linked to the intrinsic discrete nature of contact planning: advancing along the rough trajectory may require to move lightly and/or shorty away from the trajectory, what is discouraged by the shape of \( f_T \), and thus wont be the first option considered by the planner.

\[
f_T(q) = d(q, T)^2 - k \cdot l(q)
\]  

where the \( \lambda_i \) are parameters whose values are 1 or 0 depending on whether we take into account the \( i \)th individual potential field or not. Typically we take \( \lambda_i = 1 \) for the feet and hands and sometimes for the knees. This methods allows the selected bodies to anticipate by being ahead of the overall motion: since advancing along a trajectory gives a bonus, this compensate for a biggest yet meaningful gap between the posture found and the reference posture given by \( T \). Effectiveness of this approach is shown hereafter.

V. SIMULATION EXAMPLES

In our previous work ([5]), demonstration was made of the planner to cope with heavily constrained spaces. But the robot was beginning inside a tight space. We were thus legitimately asked about the capacity of our planner to drive the robot so as to enter such constrained spaces. We choose scenarios here that demonstrate this capacity, which, as we will see, is only effectively achieved with our second contribution.

A. Move and sit on a chair at a table while holding a filled glass

Our first scenario aims at demonstrating the inclusion of an additional task in the planning. To do so we take the scenario we describe in section III, namely carrying a filled glass without spilling a drop of liquid, and we merge it with the planning of sitting on a chair at a table. This planning is difficult since it imposes the robot to enter a narrow space.

Planning is successfully achieved in about 4 hours during which 5400 nodes are generated. The output plan consists in 69 nodes, some of which are depicted in Fig. 5. The robot is first walking, then by helping with his left hand it finally manages to place its left foot in front on the chair (around the 50th node). At this stage it has found an entry point into the narrow space between the table and the chair and begins to move on the chair with the help of its thighs. The same scenario using the modified potential field of section IV is done in about 3 hours and 3800 nodes.

B. Passing under a table

Our second scenario emphasizes on the gain due to individual potential fields. Passing under a table requires to cope with both the very small steps of four-legged motion and, above all, the change of posture types, which happens twice: the robot stands at first in front of the table, must then pass under it, and finishes in an upright posture on the other side. The difficulties are illustrated by Fig. 6. Both figures show the free flyer (i.e. waist) position for every valid scenario we describe in section III, namely carrying a filled glass without spilling a drop of liquid, and we merge it with the planning of sitting on a chair at a table. This planning is difficult since it imposes the robot to enter a narrow space.

Planning is successfully achieved in about 4 hours during which 5400 nodes are generated. The output plan consists in 69 nodes, some of which are depicted in Fig. 5. The robot is first walking, then by helping with his left hand it finally manages to place its left foot in front on the chair (around the 50th node). At this stage it has found an entry point into the narrow space between the table and the chair and begins to move on the chair with the help of its thighs. The same scenario using the modified potential field of section IV is done in about 3 hours and 3800 nodes.
VI. EXPERIMENT CASES STUDY WITH HRP-2

Using the framework described in [5], we played the output of the glass scenario on HRP-2 robot. Since some posture were really demanding for the robot in term of torques, and some steps redundant, we manually removed some nodes. We were able to play successfully the plan with a glass filled with Japanese tea. Some shots of this experiment are given by Fig. 8, and the attached video shows the planning workflow and the final trajectory executed by the HRP-2 robot.

VII. CONCLUSION

In this work we presented several extensions to our planner. Namely, how to cope with additional tasks and an improvement of the potential field function we used that allow us to plan faster and in more difficult scenario. Some more works remain. First, the planner tends to propose redundant contacts sets, or contacts that are too close. Second, it would be cleaner to take torque limitations into account at the contact planning level, whereas they are currently dealt with during the post-processing. We are working on these issue, especially we are looking at automating contacts sets filtering.

REFERENCES


Fig. 7. Passing under the table using individual trajectories for hands, feet and knees. The total movement is made in 141 nodes (from 0 to 140). Between two pictures, there is a “time” of 10 nodes. Note that the longest part of the plan is to go from the upright posture to the four-legged one.

Fig. 8. The glass scenario played on HRP-2. The glass remains perfectly vertical during the whole motion.